STACO: STREAMING TEXT-DRIVEN HUMANOID CONTROL WITH SHARED STATE–COMMAND LATENTS

Anonymous authors
Paper under double-blind review

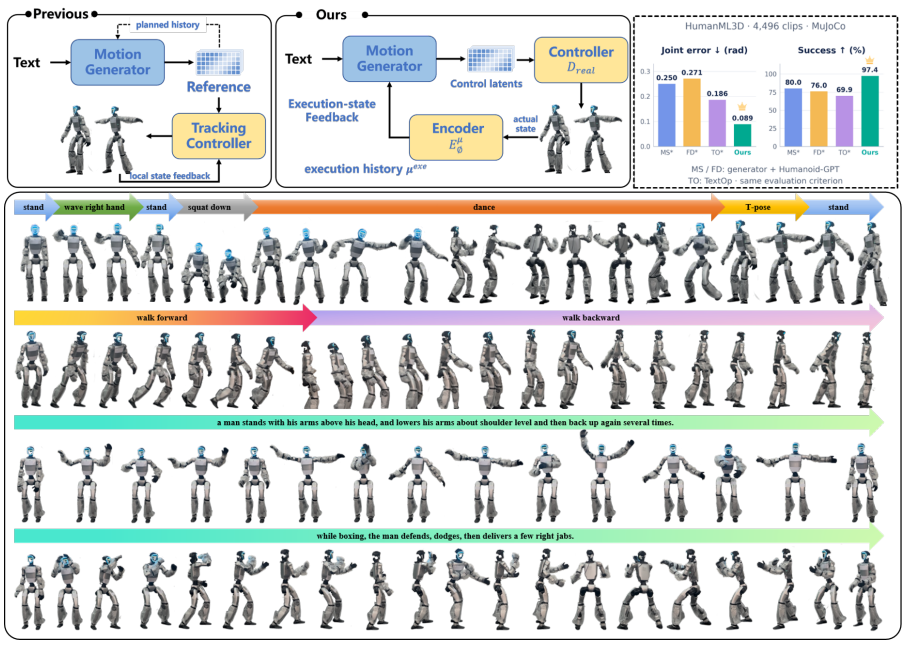

Figure 1: **StaCo** is a shared state–command latent framework for streaming language-driven humanoid control. It maps command latents directly to actions and feeds executed states back to the generator, avoiding explicit motion decoding, online retargeting, and separate trajectory tracking. The figure contrasts StaCo with reference-tracking pipelines, reports HumanML3D control results in MuJoCo, and presents representative continuous whole-body behaviors.

ABSTRACT

Existing latent-driven methods for text-driven humanoid control streamline the generation-to-control pipeline, but typically learn motion representations independently of control policies and rely on command-centric latents. This separation yields representations that are weakly aligned with robot execution and prevents the generator from incorporating the robot's actual execution state into future predictions. We propose **StaCo**, a shared state–command latent framework that couples motion representation learning with humanoid control and closes the feedback loop between streaming generation and physical execution. StaCo trains a Control-Native Motion VAE through teacher–student policy distillation, yielding a robot-native latent space that serves both as an execution-state representation and as an executable control target. Built on this shared latent space, a streaming DiT predicts future command latents from language and historical execution latents, while proprioceptive feedback is re-encoded after execution to inform subsequent generation. Experiments demonstrate that the learned latent space supports robust humanoid control while retaining high-fidelity motion structure, and that the streaming generator produces high-quality motions consistent with language instructions. End-to-end evaluation achieves a 97.35% strict control success rate in MuJoCo, while text-conditioned generation attains an FID of 19.14 on HumanML3D-derived robot motions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Recent advances in text-driven human motion generation have enabled semantically rich and diverse behaviors to be synthesized directly from natural-language descriptions Zhang et al. (2023); Jiang et al. (2023); Guo et al. (2024); Barquero et al. (2024). Extending this capability to humanoid robots opens a promising avenue toward more natural and responsive human–robot interaction Shao et al. (2025); Xie et al. (2026). Unlike offline human motion synthesis, however, real-time humanoid interaction requires robots to continuously translate streaming language instructions into coordinated and physically executable whole-body behaviors while adapting to their evolving execution state.

Existing systems typically bridge motion generation and robot control through intermediate motion representations. One line of work generates human motion from text and transfers the resulting trajectories to humanoid embodiments through retargeting Li et al. (2025b); Xiao et al. (2025); Cai et al. (2026); Araujo et al. (2025), while another directly predicts robot trajectories or joint-space targets that are executed by downstream tracking controllers Jiang et al. (2025); Xie et al. (2026); Huang et al. (2026); Ji et al. (2024); He et al. (2024); Ze et al. (2025); Liao et al. (2026). More recently, latent-driven approaches have begun to streamline this interface by predicting compact motion or control latents that directly condition low-level policies Li et al. (2026); Yuan et al. (2026); Jia et al. (2026). By bypassing explicit motion decoding and online retargeting, these approaches substantially shorten the generation-to-control pipeline.

Despite this progress, existing interfaces remain limited in two important respects. First, explicit-motion pipelines introduce multiple transformations between high-level generation and physical execution, including motion decoding, embodiment adaptation, and trajectory tracking. These sequential stages increase latency and create multiple sources of mismatch between the generated motion and the robot's realized behavior. Latent-driven methods alleviate much of this interface overhead, but their latent spaces are commonly learned by standalone motion autoencoders optimized primarily for reconstruction Li et al. (2026); Yuan et al. (2026), with representation learning and control optimization treated as separate stages. The resulting latent space is therefore not explicitly aligned with robot execution. Second, existing latents are predominantly command-centric: they encode intended motion but do not represent the robot's actual post-execution state in the same space. When tracking errors, disturbances, or behavior transitions cause execution to diverge from intent, this discrepancy is not propagated back to the high-level generator. Consequently, low-level control may operate in closed loop, while future motion generation remains conditioned primarily on previously generated intentions rather than the state actually reached by the robot.

To overcome these limitations, we introduce **StaCo**, a shared state–command latent framework for streaming text-driven humanoid control. StaCo learns a control-native latent space that simultaneously represents the robot's execution state and serves as an executable control target, thereby coupling motion representation learning with humanoid control. A Control-Native Motion VAE is trained through teacher–student policy distillation, jointly learning a robot-native state encoder and a latent-conditioned control decoder under executable action supervision. Built on this shared latent space, a streaming DiT predicts future command latents from language and historical execution latents, while proprioceptive feedback is re-encoded after execution and fed back into subsequent generation. This shared state–command interface closes the loop between high-level generation and physical execution, allowing future commands to adapt to what the robot actually executes rather than only to what the generator previously intended. Experiments demonstrate that StaCo achieves 97.35% strict control success in MuJoCo while attaining an FID of 19.14 for text-conditioned generation on HumanML3D-derived robot motions, validating its ability to jointly support reliable execution and high-quality language-conditioned motion generation.

Our contributions are threefold:

- **Unified generation–control latent interface.** We introduce a robot-native latent interface shared by the streaming motion generator and low-level controller, avoiding explicit motion decoding, online retargeting, and separate trajectory tracking during deployment.
- **Control-Native Motion VAE.** We jointly optimize a shared state encoder and latent-conditioned control decoder through teacher–student control distillation, yielding latents that represent robot motion states and directly condition executable actions.
- **Execution-aware streaming DiT.** We develop a streaming DiT that predicts future command latents from language instructions and re-encoded execution history, closing the loop between high-level motion generation and physical execution.

2 Related Work

2.1 Explicit-Motion Interfaces for Text-Driven Humanoid Control

Text-driven humanoid systems commonly connect generation and control through explicit kinematic motion representations. MotionStreamer and FloodDiffusion generate human motion from language and subsequently transfer it to humanoid embodiments through retargeting methods such as GMR Xiao et al. (2025); Cai et al. (2026); Araujo et al. (2025). Within this pipeline, GENMO improves the generalization of text-conditioned motion generation, while SONIC develops a generalist controller for tracking diverse motion references Li et al. (2025a); Luo et al. (2026). Robot-native methods reduce embodiment-transfer requirements by directly generating robot-space references: RobotMDM and UniAct predict joint-space trajectories, whereas OMG and TextOp support multimodal or autoregressive robot-motion generation Serifi et al. (2024); Jiang et al. (2025); Huang et al. (2026); Xie et al. (2026). PhyGile and SafeFlow further improve physical feasibility through motion refinement or safety filtering Bao et al. (2026); Cho et al. (2026). Nevertheless, these approaches still employ explicit motion trajectories as the interface between generation and execution, requiring retargeting or feasibility processing when necessary and downstream controllers for physical tracking. Such multi-stage pipelines introduce additional latency and may accumulate discrepancies between generated and executed motions.

2.2 Latent and Direct-Action Interfaces for Text-Driven Humanoid Control

Recent approaches shorten the generation-to-execution pipeline by reducing or bypassing explicit motion references. Direct-action methods such as LangWBC predict actions from language and proprioceptive history, while SENTINEL and SCRIPT directly generate action or state–action chunks from physical observations Shao et al. (2025); Wang et al. (2025); Zhang et al. (2026). Although these methods tightly couple high-level generation with execution, they do not expose a reusable motion representation that can support both motion modeling and control. Hierarchical approaches instead retain a compact latent interface: RoboGhost conditions a diffusion action policy on motion latents, RoboForge combines reconstruction-pretrained motion latents with tracking through offline optimization, and DAJI learns a control-oriented joint-intent representation Li et al. (2026); Yuan et al. (2026); Jia et al. (2026). These methods demonstrate that latent representations can substantially shorten the interface between motion generation and humanoid control. However, existing latent interfaces primarily represent intended motion and generally do not re-encode the robot's post-execution state into the same space to condition subsequent generation. Consequently, execution feedback remains confined to the low-level controller rather than being propagated into the high-level generative process.

3 Method

3.1 Overview

We introduce **StaCo**, a streaming text-driven humanoid control framework built on a shared state–command latent space. Its central principle is to represent motion commands and observed execution states within a unified robot-native space, providing a common interface for high-level motion generation and low-level control. As illustrated in Fig. 2, StaCo comprises two core components, i.e., a Control-Native Motion VAE that jointly learns a motion-state encoder and a latent-conditioned control decoder, and a streaming text-conditioned generator that predicts future command latents. The shared encoder additionally maps the robot's actual execution states into this latent space, enabling execution feedback to directly condition subsequent generation.

To establish the control-native latent space, a privileged motion-tracking teacher generates action targets conditioned on reference motions and the robot's current state. Distilling these actions jointly shapes the encoder and control decoder, aligning the learned motion representation with the requirements of physical execution. Built on this representation, the streaming generator predicts future command latents from language instructions and encoded execution history. During deployment, the control decoder converts generated latents into robot actions using proprioceptive history, while observed execution states are continuously re-encoded to inform subsequent predictions. This formulation bypasses explicit motion reconstruction, online retargeting, and a separate reference-tracking

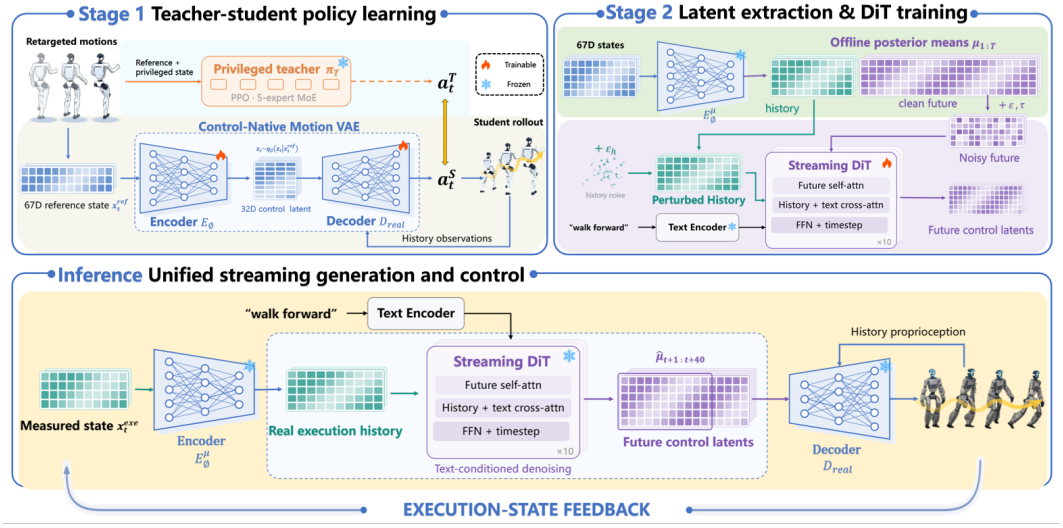

Figure 2: **Overview of StaCo.** We propose a closed-loop framework that connects text-conditioned motion generation and humanoid control through a shared state–command latent space, with executed robot states fed back to guide subsequent predictions.

module at inference time, and closes the feedback loop between language-conditioned generation and physical execution.

3.2 Motion Control with Shared State–Command Latents

Motion representations learned independently of control may not provide suitable command conditions for a dynamically constrained humanoid. We address this limitation through teacher–student learning, in which action supervision jointly shapes a variational motion encoder and a latent-conditioned control decoder. A privileged teacher first learns to track robot-native reference motions, and the student subsequently distills its control behavior through the shared latent representation.

3.2.1 Privileged Teacher Policy

Following RoboGhost Li et al. (2026), we train a privileged motion-tracking teacher policy π_T using PPO Schulman et al. (2017). Human motion sequences are first retargeted to the robot's 29-DoF joint space using GMR and resampled from 30 Hz to 50 Hz. The teacher policy $\pi_T(\mathbf{a}_t^T \mid \mathbf{o}_t^T)$ accesses privileged simulation states during training and tracks only the current reference frame. Its observation is defined as

$$\mathbf{o}_t^T = [\mathbf{q}_t^{\text{ref}}, \dot{\mathbf{q}}_t^{\text{ref}}, \Delta\mathbf{p}_t^{\text{anc}}, \Delta\mathbf{r}_t^{\text{anc}}, \Delta\mathbf{q}_t, \Delta\mathbf{p}_t^{\text{key}}, \mathbf{s}_t^{\text{pro}}], \tag{1}$$

where the tracking-error terms are computed using ground-truth anchor poses, joint states, and key rigid-body positions available in simulation. The proprioceptive state is defined as $\mathbf{s}_t^{\text{pro}} = [{}^{\mathcal{B}}\boldsymbol{\omega}_t, \mathbf{q}_t^{\text{rel}}, \dot{\mathbf{q}}_t, \mathbf{a}_{t-1}] \in \mathbb{R}^{90}$, comprising base angular velocity, 29-DoF joint positions relative to the default pose, joint velocities, and the previous action. We adopt asymmetric actor–critic training, in which the critic additionally accesses privileged quantities such as base linear velocity and full rigid-body poses. These additional quantities are not exposed to the subsequent student policy. The teacher outputs normalized joint actions $\mathbf{a}_t^T \in \mathbb{R}^{29}$.

To accommodate diverse motion modes, the teacher actor employs a mixture-of-experts (MoE) architecture with five experts. A gating network predicts expert weights $g_m(\mathbf{o}_t^T)$ from the current observation, and the final action is computed as $\mathbf{a}_t^T = \sum_{m=1}^{5} g_m(\mathbf{o}_t^T)\pi_m(\mathbf{o}_t^T)$. After training, we deterministically replay all candidate motion sequences with the teacher and discard those that terminate because of excessive anchor-height or body-pose errors. The retained sequences form a set of robot-native motions that can be reliably executed by the teacher and are subsequently used for student-policy and motion-generator training.

3.2.2 CONTROL-NATIVE MOTION VAE STUDENT POLICY

To learn a shared state–command latent space that is both motion-expressive and directly controllable, we construct a Control-Native Motion VAE. The encoder E_ϕ takes a 67-D motion representation compatible with the robot's online state measurements, $\mathbf{x}_t = [{}^{\mathcal{T}}\boldsymbol{\omega}_t, \rho_{6D}(\mathbf{R}_t), \mathbf{q}_t, \dot{\mathbf{q}}_t] \in \mathbb{R}^{67}$, comprising local angular velocity, a 6-D representation of root orientation, and 29-DoF joint positions and velocities. This representation can be constructed from either retargeted reference motions or the robot's proprioceptive state, allowing the same encoder to map motion targets and actual execution states into a shared latent space. The encoder parameterizes a Gaussian posterior, from which the latent is sampled through reparameterization:

$$q_\phi(\mathbf{z}_t \mid \mathbf{x}_t) = \mathcal{N}(\boldsymbol{\mu}_t, \mathrm{diag}(\exp \boldsymbol{\lambda}_t)), \qquad \mathbf{z}_t = \boldsymbol{\mu}_t + \exp\left(\tfrac{1}{2}\boldsymbol{\lambda}_t\right) \odot \boldsymbol{\epsilon}_t, \tag{2}$$

where $\boldsymbol{\mu}_t$ and $\boldsymbol{\lambda}_t$ denote the posterior mean and log-variance, respectively, and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

The control decoder D_{real} is a residual MLP conditioned on the motion latent and a 15-frame proprioceptive history $\mathbf{h}_t^{\mathrm{pro}} \in \mathbb{R}^{1350}$, which stacks base angular velocity, relative joint positions, joint velocities, and previous actions. The student action is given by $\mathbf{a}_t^S = D_{\mathrm{real}}(\mathbf{z}_t, \mathbf{h}_t^{\mathrm{pro}}) \in \mathbb{R}^{29}$. Gradients from teacher-action supervision propagate through both D_{real} and E_ϕ, allowing control requirements to directly shape the latent representation. The resulting latents encode target motion states while serving as command conditions for the low-level controller, providing a compact interface between motion generation and robot control.

We train the student through DAgger-style online distillation Ross et al. (2011). During student rollouts in simulation, the frozen privileged teacher provides target actions at the states actually visited by the student. The overall objective is

$$\mathcal{L}_{\mathrm{VAE}} = \mathcal{L}_{\mathrm{ctrl}} + \lambda_{KL}\mathcal{L}_{\mathrm{KL}} + \lambda_z\mathcal{L}_z, \tag{3}$$

where $\mathcal{L}_{\mathrm{ctrl}}$ is the Huber loss between the student and teacher actions, $\mathcal{L}_{\mathrm{KL}}$ regularizes the posterior toward a standard Gaussian prior, and $\mathcal{L}_z$ constrains the latent magnitude. We use sampled latents $\mathbf{z}_t$ during student training. Once the model is frozen, motion-generator training and deployment use the posterior mean $\boldsymbol{\mu}_t$ as the deterministic latent representation. At inference time, the teacher is removed, and D_{real} converts generated command latents into robot actions conditioned on the current proprioceptive history.

3.3 STREAMING TEXT-TO-COMMAND LATENT GENERATION

Building on the control-native latent space, we formulate text-driven motion generation as conditional latent trajectory prediction. We freeze the motion encoder E_ϕ and use its posterior mean $\boldsymbol{\mu}_t \in \mathbb{R}^{32}$ as the deterministic representation of each motion frame. Given a text instruction c_t, the generator predicts future command latents conditioned on the most recent $L = 20$ latent frames, $\mathbf{H}_t = \boldsymbol{\mu}_{t-L:t-1} \in \mathbb{R}^{L\times 32}$. Specifically, it models $p_\theta(\mathbf{X}_t \mid \mathbf{H}_t, c_t)$, where $\mathbf{X}_t = \boldsymbol{\mu}_{t:t+H-1} \in \mathbb{R}^{H\times 32}$ is the target sequence of $H = 40$ future command latents. At a control frequency of $50\,\mathrm{Hz}$, the history and prediction windows correspond to $0.4\,\mathrm{s}$ and $0.8\,\mathrm{s}$, respectively. The predicted latents directly condition the control decoder D_{real}, eliminating the need for explicit joint-trajectory reconstruction, online motion retargeting, or a separate trajectory-tracking stage.

Conditional diffusion transformer. We model the conditional distribution using a future-only Diffusion Transformer (DiT). A frozen Sentence-T5 Raffel et al. (2020) encoder maps the text instruction to semantic features $\mathbf{H}_t^{\mathrm{text}}$. Historical latents are processed by a dedicated temporal projection and augmented with positional, valid-frame, and instruction-assignment embeddings. The instruction-assignment embeddings indicate whether each historical frame corresponds to the current or preceding instruction. The encoded latent history and text features form the conditional memory, while noisy future latents form the DiT query sequence. Each Transformer block applies self-attention among future tokens to capture temporal dependencies, followed by cross-attention to the conditional memory. The diffusion timestep τ modulates the intermediate features. Let $\mathbf{X}_t^0 = \mathbf{X}_t$ denote the clean future latent trajectory. The forward diffusion process is

$$\mathbf{X}_t^\tau = \sqrt{\bar{\alpha}_\tau}\,\mathbf{X}_t^0 + \sqrt{1-\bar{\alpha}_\tau}\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{4}$$

We adopt x_0-prediction, with the generator estimating the clean trajectory as $\widehat{\mathbf{X}}_t^0 = G_\theta\left(\mathbf{X}_t^\tau, \tau, \widetilde{\mathbf{H}}_t, \mathbf{H}_t^{\mathrm{text}}\right)$, where $\widetilde{\mathbf{H}}_t$ denotes the augmented historical condition defined below.

The training objective combines Min-SNR-weighted denoising with temporal and boundary regularization:

$$\mathcal{L}_{\text{gen}} = \mathcal{L}_{x_0} + \lambda_v \mathcal{L}_{\text{vel}} + \lambda_a \mathcal{L}_{\text{acc}} + \lambda_b \mathcal{L}_{\text{bnd}}, \tag{5}$$

where $\mathcal{L}_{x_0}$ is the Min-SNR-weighted denoising loss, and $\mathcal{L}_{\text{vel}}$ and $\mathcal{L}_{\text{acc}}$ encourage consistency of first- and second-order temporal differences in latent space, respectively. The boundary loss $\mathcal{L}_{\text{bnd}}$ constrains both the first predicted latent and its transition from the final historical latent, promoting continuity across generation windows.

Execution-aware history augmentation. During training, historical conditions are encoded from reference motions, whereas during deployment they are encoded from the robot's actual execution states. To mitigate the resulting conditional distribution shift, we retain the deterministic posterior mean $\boldsymbol{\mu}$ as the future supervision target while perturbing the historical condition through posterior sampling and additive Gaussian noise:

$$\widetilde{\mathbf{H}}_t = (1-b)\mathbf{H}_t^{\mu} + b\mathbf{H}_t^{z} + \boldsymbol{\eta}, \qquad b \sim \text{Bernoulli}(p_z), \quad \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma_h^2 \mathbf{I}), \tag{6}$$

where $\mathbf{H}_t^{\mu}$ contains posterior means and $\mathbf{H}_t^{z}$ contains samples from the corresponding VAE posteriors. This augmentation exposes the generator to variations around reference-motion histories, encouraging robustness to imperfect historical conditions while preserving deterministic future targets. We further construct training windows by mixing regular motion segments, instruction-transition segments, and highly dynamic segments to prevent smooth motions from dominating training. Text conditions are randomly dropped during training to support classifier-free guidance.

3.4 Execution-State Feedback for Closed-Loop Control

Conditioning subsequent predictions solely on previously generated latents leaves discrepancies between planned and realized motion unobserved by the generator. StaCo therefore incorporates execution feedback into generation through the control-native latent space. At each control step t, the executed robot state $\mathbf{x}_t^{\text{exe}}$ is constructed from onboard IMU and joint measurements and mapped by the frozen Motion VAE encoder to its posterior mean $\boldsymbol{\mu}_t^{\text{exe}}$. The latest 20 execution latents form a rolling history $\mathbf{H}_t^{\text{exe}} = \boldsymbol{\mu}_{t-19:t}^{\text{exe}}$, which serves as the historical condition for the next generation step in place of reference or previously predicted latents. Each replan thus predicts the subsequent 40 command latents conditioned on the robot's actual execution history. Each generated latent is decoded by D_{real}, together with the live proprioceptive history, into a 29-dimensional joint action. This establishes a closed-loop hierarchy in which lower-frequency latent motion planning incorporates execution-state feedback, while high-frequency control uses proprioceptive history to generate robot actions.

4 Experiments

4.1 Experimental Setup

Datasets. We use HumanML3D Guo et al. (2022) and BABEL Punnakkal et al. (2021) as source motion datasets. Human motions are retargeted offline to the Unitree G1 using GMR Araujo et al. (2025), resampled to $50\,\text{Hz}$, and encoded as unified 67-dimensional robot-native states. We discard sequences that the privileged teacher cannot track stably or that exhibit implausible contacts. The resulting corpus contains motions that satisfy both teacher-tracking and contact-plausibility criteria and is used to train the Motion VAE and motion generator.

Metrics. We evaluate both motion tracking in physics simulators and text-conditioned motion generation. Inspired by the tracking evaluation in RoboGhost Li et al. (2026), we report two success rates to assess sequence completion and tracking accuracy. Succ_{nf} measures the proportion of sequences completed without falling or severe trajectory deviation. To additionally account for tracking accuracy, the stricter Succ requires these completed sequences to satisfy sequence-mean joint-angle and keypoint-position error thresholds of $\text{Empjpe} \leq 0.20\,\text{rad}$ and $\text{Empkpe} \leq 0.15\,\text{m}$, respectively. Tracking errors are aggregated only over complete trajectories satisfying Succ_{nf}. For text-conditioned motion generation, we follow MotionStreamer Xiao et al. (2025) and report R@1, R@2, R@3, MM-Dist, FID, and Diversity. For long-term generation on BABEL, we evaluate subsequences using R@3, FID, Diversity, and MM-Dist, and transitions using FID, Diversity, PJ, and AUJ. Further data-processing and evaluation details are provided in the appendix.

Table 1: **Physics-based motion tracking.** Motion tracking performance of the Motion VAE on the HumanML3D test set in IsaacLab and MuJoCo.

	IsaacLab				MuJoCo			
Method	$\text{Succ}_{\text{nf}}\uparrow$	Succ ↑	Empjpe↓	Empkpe↓	$\text{Succ}_{\text{nf}}\uparrow$	Succ ↑	Empjpe↓	Empkpe↓
TextOp	**92.7016**	70.2375	0.1799	0.0840	**93.1717**	70.4849	0.1800	0.0685
DAJI-Act*	88.6690	86.6403	0.1244	0.0544	88.8916	86.9866	0.1241	0.0556
Privileged teacher	87.8773	86.1207	0.1242	**0.0459**	86.9372	85.3290	0.1243	**0.0468**
Student 8D	90.9451	83.7704	0.1425	0.0605	92.0831	84.9579	0.1416	0.0614
Student 16D	89.4607	86.5661	0.1280	0.0554	90.4008	**87.3577**	0.1275	0.0563
Ours w/o teacher 32D	**94.8788**	81.1974	0.1635	0.0655	**95.4231**	82.0881	0.1626	0.0666
Separate training	89.0153	83.9931	0.1497	0.0696	89.2133	85.4033	0.1473	0.0700
Ours sampled-z	89.3122	**87.2588**	**0.1237**	**0.0543**	89.6338	**87.5309**	**0.1224**	**0.0547**
Ours μ	89.1390	**87.4072**	**0.1232**	**0.0542**	89.4112	87.2835	**0.1221**	**0.0544**

Table 2: **Latent motion reconstruction.** Quantitative evaluation of motion reconstruction on the HumanML3D test set.

Method	FID↓	R@1↑	R@2↑	R@3↑	Div→	MM-Dist↓	Empjpe↓
Ground Truth	0.0000	0.6797	0.8323	0.8891	26.1282	15.3463	–
MotionStreamer TAE	5.7427	0.6636	0.8188	0.8784	25.9891	15.5512	0.0798
FloodDiffusion VAE	39.7644	0.5612	0.7283	0.8065	25.5127	17.0394	0.1566
Student 8D	21.2174	0.6262	0.7889	0.8538	25.5270	16.0259	0.1131
Student 16D	7.9138	0.6613	0.8174	0.8759	25.8874	15.5490	0.0748
Ours w/o teacher 32D	2.8310	0.6737	0.8284	0.8865	25.9705	**15.3308**	0.0582
Separate training	14.2641	0.6502	0.8085	0.8721	25.8800	15.8128	0.0726
Ours sampled-z	**1.0681**	**0.6762**	**0.8294**	**0.8868**	**26.0643**	15.3667	**0.0358**
Ours μ	**0.9257**	**0.6759**	**0.8298**	**0.8867**	**26.0857**	**15.3398**	**0.0352**

4.2 Evaluation of Control and Reconstruction

We evaluate whether the latent space learned through teacher-guided control supports accurate physical execution while retaining detailed motion information. Control performance and reconstruction accuracy are assessed separately on robot motions retargeted from the HumanML3D test split. For control evaluation, the encoder maps reference robot states to latents, and D_{real} combines these latents with proprioceptive history to generate robot actions. As summarized in Table 1, our posterior-mean model achieves high strict success rates and low joint and keypoint tracking errors in both IsaacLab and MuJoCo, consistently outperforming TextOp Xie et al. (2026) and DAJI-Act Jia et al. (2026). Although TextOp obtains higher no-fall completion rates, its lower strict success rates indicate that motion completion does not necessarily imply accurate reference tracking.

To quantify the motion information retained in the control-trained latent space, we freeze the encoder and subsequently train an auxiliary decoder D_{recon} to reconstruct the 67-dimensional robot motion states from its latents. As shown in Table 2, our posterior-mean representation substantially outperforms the MotionStreamer TAE Xiao et al. (2025) and FloodDiffusion VAE Cai et al. (2026) in reconstruction quality, while preserving text–motion alignment and diversity close to the ground truth. These results demonstrate that the control-trained latent space retains detailed, recoverable motion information despite the absence of an explicit reconstruction objective during latent learning. The auxiliary decoder is used solely for evaluation and does not shape the control latent space or participate in robot deployment.

4.3 Evaluation of Streaming Text-to-Motion Generation

We evaluate the streaming text-conditioned latent generator on HumanML3D and BABEL, covering single-instruction generation, long-term streaming generation, and physics-based execution. For generation evaluation, the auxiliary decoder D_{recon} converts predicted latents into 67-dimensional robot motions solely for metric computation. We compare against MotionStreamer Xiao et al. (2025) and FloodDiffusion Cai et al. (2026) followed by robot retargeting, the robot-native TextOp Xie et al. (2026), and the latent-driven DAJI-Flow Jia et al. (2026).

As shown in Table 3, our method achieves the best overall generation performance on HumanML3D, demonstrating improved distributional fidelity, text–motion alignment, and motion diversity. For long-term generation on BABEL, it also achieves the strongest subsequence-level semantic align-

Table 3: **Text-conditioned robot-motion generation.** Quantitative results on the HumanML3D and BABEL test sets for single-instruction and long-term streaming generation, respectively.

(a) Text-conditioned robot-motion generation on HumanML3D.

Method	FID↓	R@1↑	R@2↑	R@3↑	Div→	MM-Dist↓
Ground Truth	0.0000	0.6797	0.8323	0.8891	26.1282	15.3463
MotionStreamer*	36.8578	0.4522	0.6047	0.6906	**25.4382**	18.1949
FloodDiffusion*	109.8693	0.2321	0.3808	0.4837	24.7270	20.9644
TextOp*	**36.3538**	**0.5679**	**0.7446**	**0.8254**	25.3517	**16.6467**
DAJI-Flow*	124.5287	0.4598	0.6185	0.7092	22.8148	18.7587
Ours	**19.1432**	**0.5875**	**0.7589**	**0.8315**	**26.0769**	**16.3955**

(b) Long-term motion generation on BABEL.

Method	Subsequence				Transition			
	R@3↑	FID↓	Div→	MM-Dist↓	FID↓	Div→	PJ→	AUJ↓
Ground Truth	0.4693	0.0000	24.6936	19.8129	0.0000	22.3288	0.0124	0.0000
MotionStreamer*	0.3689	**31.0224**	**23.4553**	20.7207	**19.2542**	**21.3857**	0.1169	2.8484
FloodDiffusion*	0.2681	94.9903	23.2898	21.7897	78.4466	21.3531	0.1166	2.9299
TextOp*	0.4016	67.2522	20.4705	**20.0268**	59.3871	17.1760	**0.0152**	**0.0965**
DAJI-Flow*	**0.4257**	97.0896	23.4355	20.4595	66.8344	20.5342	0.2903	6.1244
Ours	**0.4371**	**26.7476**	**25.0570**	**20.4296**	**20.2987**	**22.9763**	**0.0365**	**0.4847**

Table 4: **End-to-end humanoid control.** Physics-based execution performance on the HumanML3D test set in MuJoCo, measured by control success and motion-tracking errors.

Method	Succ↑	Empjpe↓	Empkpe↓
MotionStreamer* + Humanoid-GPT Qi et al. (2026)	80.0489	0.2499	0.1318
FloodDiffusion* + Humanoid-GPT Qi et al. (2026)	76.0009	0.2709	0.1452
TextOp*	69.8621	0.1864	0.1437
Ours w/o history perturbation	97.0863	**0.0894**	**0.0561**
Ours w/o execution feedback	**97.9537**	0.1090	0.0580
Ours	**97.3532**	**0.0889**	**0.0551**

ment and distributional quality. At instruction transitions, our method maintains smooth motion continuity while preserving a motion distribution close to the ground truth, providing a favorable balance between transition smoothness and generation quality.

For physics-based execution, the generated latents directly condition D_{real} during closed-loop simulation in MuJoCo, without involving D_{recon} in the control loop. As summarized in Table 4, our full model achieves the highest strict success rate and the lowest joint and keypoint tracking errors among the compared methods. These results show that the generated latents support both semantically aligned motion generation and accurate physical execution through a shared control interface.

4.4 Ablation Studies

Latent Space and Joint Training. Tables 1 and 2 show that increasing the latent dimensionality from 8 to 32 substantially improves reconstruction quality and reduces tracking errors. These results suggest that the 8-D bottleneck restricts the motion information available for reconstruction and control. Removing teacher supervision increases completion rates under Succ_{nf} but lowers strict success and increases tracking errors, supporting the role of teacher supervision in promoting accurate motion execution beyond stable completion. Separate training also yields higher reconstruction and tracking errors and lower strict success than joint training. This is consistent with the limitation of learning a reconstruction-oriented representation that remains fixed during controller training. Joint optimization allows control supervision to shape the encoder directly, providing a latent representation that supports both motion reconstruction and accurate control.

Execution-State Feedback and History Augmentation. Table 4 evaluates the contributions of execution-state feedback and history perturbation. The no-feedback variant reuses previously generated control latents as history without incorporating the robot's actual execution states into sub-

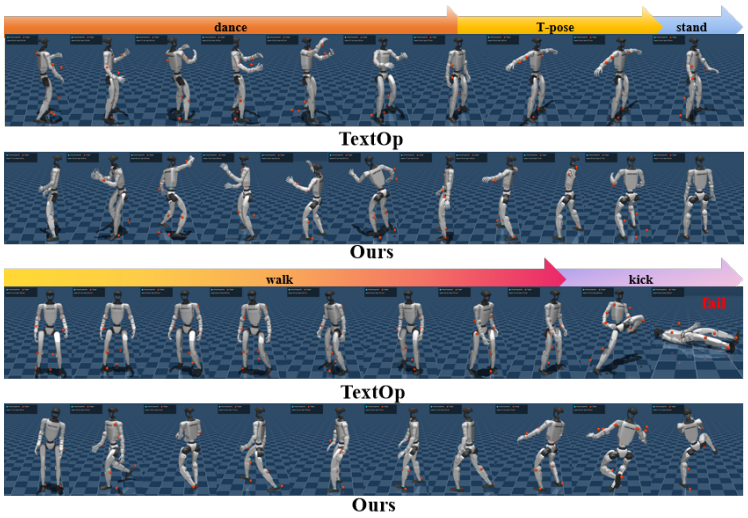

Figure 3: **Simulation-based quantitative comparison.** In MuJoCo, the Empkpe values for the first and second motion clips are 0.0645 m and 0.0578 m for TextOp, and 0.0382 m and 0.0364 m for Ours, respectively.

Table 5: Ablation study of the streaming motion generator.

Method	FID↓	R@1↑	R@2↑	R@3↑	Div→	MM-Dist↓
Ground Truth	0.0000	0.6797	0.8323	0.8891	26.1282	15.3463
CLIP Radford et al. (2021)	24.4858	0.4995	0.6625	0.7509	25.9058	17.4664
w/o mixed motion windows	26.4446	0.5208	0.6940	0.7857	25.6525	17.2357
Ours sampled-z	**18.3153**	**0.5873**	**0.7544**	**0.8270**	**25.8295**	**16.4047**
Ours	**19.1432**	**0.5875**	**0.7589**	**0.8315**	**26.0769**	**16.3955**

sequent generation. In contrast, the full model re-encodes the latest executed state and updates the generation history at every replanning step. Although the two variants achieve comparable strict success rates, execution feedback yields lower joint and keypoint tracking errors, indicating improved tracking precision. History perturbation provides a further consistent improvement by enhancing robustness to discrepancies between generated and executed motions.

Generator Design. Table 5 shows that Sentence-T5 Raffel et al. (2020) achieves better text–motion alignment than CLIP Radford et al. (2021), supporting its use as the text encoder in our framework. Mixed window sampling combines regular motion segments, instruction-transition segments, and highly dynamic segments to increase the training coverage of transitions and dynamic behaviors beyond steady motions. Removing this strategy degrades both distributional quality and text–motion alignment, indicating that the composition of training windows is important for streaming motion generation.

5 Conclusion

We presented StaCo, a shared state–command latent framework for streaming language-conditioned humanoid control. Through teacher–student policy distillation, the Control-Native Motion VAE learns a robot-native latent space shared by execution-state representations and control targets, while retaining sufficient information for motion reconstruction. The streaming generator predicts future command latents from language and the robot's actual execution history, avoiding explicit motion decoding and online retargeting during deployment. Experiments across motion reconstruction, text-conditioned generation, and physics-based execution demonstrate competitive generation quality and robust humanoid control. Moreover, feeding re-encoded execution states back to the generator reduces state drift and error accumulation during closed-loop operation. Future work will extend this framework toward scene-aware, object-interactive, and longer-horizon humanoid behaviors.

6 THE USE OF LARGE LANGUAGE MODELS (LLMS)

LLMs were used in the preparation of this work solely for the purpose of grammar correction and proofreading. The LLM did not contribute to the ideation, scientific content, writing, or analysis of the research. The authors take full responsibility for the entire intellectual content of this paper.

REFERENCES

Joao Pedro Araujo, Yanjie Ze, Pei Xu, Jiajun Wu, and C Karen Liu. Retargeting matters: General motion retargeting for humanoid motion tracking. *arXiv preprint arXiv:2510.02252*, 2025.

Jiacheng Bao, Haoran Yang, Yucheng Xin, Junhong Liu, Yuecheng Xu, Han Liang, Pengfei Han, Xiaoguang Ma, Dong Wang, and Bin Zhao. Phygile: Physics-prefix guided motion generation for agile general humanoid motion tracking. *arXiv preprint arXiv:2603.19305*, 2026.

German Barquero, Sergio Escalera, and Cristina Palmero. Seamless human motion composition with blended positional encodings. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 457–469. IEEE, 2024.

Yiyi Cai, Yuhan Wu, Kunhang Li, You Zhou, Bo Zheng, and Haiyang Liu. Flooddiffusion: Tailored diffusion forcing for streaming motion generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2295–2304, 2026.

Hanbyel Cho, Sang-Hun Kim, Jeonguk Kang, and Donghan Koo. Safeflow: Real-time text-driven humanoid whole-body control via physics-guided rectified flow and selective safety gating. *arXiv preprint arXiv:2603.23983*, 2026.

Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5142–5151. IEEE, 2022.

Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. Momask: Generative masked modeling of 3d human motions. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1900–1910. IEEE, 2024.

Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.

Siqiao Huang, Kun-Ying Lee, Dongming Qiao, Guanqi He, Zhenyu Wang, Yitang Li, Shaoting Zhu, and Hang Zhao. Omg: Omni-modal motion generation for generalist humanoid control. *arXiv preprint arXiv:2606.10340*, 2026.

Mazeyu Ji, Xuanbin Peng, Fangchen Liu, Jialong Li, Ge Yang, Xuxin Cheng, and Xiaolong Wang. Exbody2: Advanced expressive humanoid whole-body control. *arXiv preprint arXiv:2412.13196*, 2024.

Haozhe Jia, Honglei Jin, Yuan Zhang, Youcheng Fan, Shaofeng Liang, Lei Wang, Shuxu Jin, Kuimou Yu, Zinuo Zhang, Jianfei Song, et al. Before the body moves: Learning anticipatory joint intent for language-conditioned humanoid control. *arXiv preprint arXiv:2605.14417*, 2026.

Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. Motiongpt: Human motion as a foreign language. *Advances in Neural Information Processing Systems*, 36:20067–20079, 2023.

Nan Jiang, Zimo He, Wanhe Yu, Lexi Pang, Yunhao Li, Hongjie Li, Jieming Cui, Yuhan Li, Yizhou Wang, Yixin Zhu, et al. Uniact: Unified motion generation and action streaming for humanoid robots. *arXiv preprint arXiv:2512.24321*, 2025.

Jiefeng Li, Jinkun Cao, Haotian Zhang, Davis Rempe, Jan Kautz, Umar Iqbal, and Ye Yuan. Genmo: A generalist model for human motion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11766–11776. IEEE, 2025a.

Zhe Li, Weihao Yuan, Yisheng He, Lingteng Qiu, Shenhao Zhu, Xiaodong Gu, Weichao Shen, Yuan Dong, Zilong Dong, and Laurence Yang. Lamp: Language-motion pretraining for motion generation, retrieval, and captioning. In *International Conference on Learning Representations*, volume 2025, pp. 84238–84250, 2025b.

Zhe Li, Yangyang Wei, Boan Zhu, Yibo Peng, Tao Huang, Pengwei Wang, Zhongyuan Wang, Cheng Chi, Chang Xu, and Shanghang Zhang. From language to locomotion: Retargeting-free humanoid control via motion latent guidance. In *International Conference on Learning Representations*, volume 2026, pp. 129292–129319, 2026.

Qiayuan Liao, Takara E Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C Karen Liu. Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion. *Science Robotics*, 11(117):eadx8924, 2026.

Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Fernando Castañeda, Sirui Chen, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, et al. Sonic: Supersizing motion tracking for natural humanoid whole-body control. *Science Robotics*, 11(117):eaed4592, 2026.

Abhinanda R Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J Black. Babel: Bodies, action and behavior with english labels. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 722–731. IEEE, 2021.

Zekun Qi, Xuchuan Chen, Dairu Liu, Chenghuai Lin, Yunrui Lian, Sikai Liang, Zhikai Zhang, Yu Guan, Jilong Wang, Wenyao Zhang, et al. Humanoid-gpt: Scaling data and structure for zero-shot motion tracking. *arXiv preprint arXiv:2606.03985*, 2026.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Agon Serifi, Ruben Grandia, Espen Knoop, Markus Gross, and Moritz Bächer. Robot motion diffusion model: Motion generation for robotic characters. In *SIGGRAPH asia 2024 conference papers*, pp. 1–9, 2024.

Yiyang Shao, Xiaoyu Huang, Bike Zhang, Qiayuan Liao, Yuman Gao, Yufeng Chi, Zhongyu Li, Sophia Shao, and Koushil Sreenath. Langwbc: Language-directed humanoid whole-body control via end-to-end learning. *arXiv preprint arXiv:2504.21738*, 2025.

Yuxuan Wang, Haobin Jiang, Shiqing Yao, Ziluo Ding, and Zongqing Lu. Sentinel: A fully end-to-end language-action model for humanoid whole body control. *arXiv preprint arXiv:2511.19236*, 2025.

Lixing Xiao, Shunlin Lu, Huaijin Pi, Ke Fan, Liang Pan, Yueer Zhou, Ziyong Feng, Xiaowei Zhou, Sida Peng, and Jingbo Wang. Motionstreamer: Streaming motion generation via diffusion-based autoregressive model in causal latent space. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10086–10096. IEEE, 2025.

Weiji Xie, Jiakun Zheng, Jinrui Han, Jiyuan Shi, Weinan Zhang, Chenjia Bai, and Xuelong Li. Textop: Real-time interactive text-driven humanoid robot motion generation and control. *arXiv preprint arXiv:2602.07439*, 2026.

Xichen Yuan, Zhe Li, Bofan Lyu, Kuangji Zuo, Yanshuo Lu, Gen Li, and Jianfei Yang. Roboforge: Physically optimized text-guided whole-body locomotion for humanoids. *arXiv preprint arXiv:2603.17927*, 2026.

Yanjie Ze, Siheng Zhao, Weizhuo Wang, Angjoo Kanazawa, Rocky Duan, Pieter Abbeel, Guanya Shi, Jiajun Wu, and C Karen Liu. Twist2: Scalable, portable, and holistic humanoid data collection system. *arXiv preprint arXiv:2511.02832*, 2025.

Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14730–14740, 2023.

Jingyan Zhang, Han Liang, Ruichi Zhang, Bin Li, Juze Zhang, Xin Chen, Jingya Wang, Lan Xu, and Jingyi Yu. Script: Scalable diffusion policy with multi-stage training for language-driven physics-based humanoid control. *arXiv preprint arXiv:2605.22894*, 2026.

A APPENDIX

Appendix overview. This appendix complements the method and experiments with details needed to reproduce StaCo. Section A.1 specifies the representations and architectures. Section A.2 describes data preparation and policy and generator training. Section A.3 distinguishes representation, generation, and physical-execution evaluation, including the BABEL streaming protocol. Section A.4 describes cross-simulator transfer and asynchronous hardware deployment. Sections A.5 and A.6 provide supplementary analyses and precise ablation definitions. Section A.7 summarizes the training and inference workflows.

A.1 IMPLEMENTATION DETAILS

Robot-native state representation. StaCo controls the 29 actuated joints of Unitree G1 at 50 Hz. The motion encoder receives

$$\mathbf{x}_t = [\,{}^{\mathcal{T}}\boldsymbol{\omega}_t, \rho_{6\mathrm{D}}(\widetilde{\mathbf{R}}_t), \mathbf{q}_t, \dot{\mathbf{q}}_t\,] \in \mathbb{R}^{67}, \tag{7}$$

where $\mathcal{T}$ denotes the torso frame. The components contain torso-local angular velocity (3), torso orientation (6), joint positions (29), and joint velocities (29). The encoder uses the same joint ordering and coordinate conventions for retargeted references and measured execution states.

For the state–command interface, orientation is expressed relative to the previous frame's heading:

$$\widetilde{\mathbf{R}}_t = \mathbf{R}_z(-\psi_{t-1})\mathbf{R}_t, \tag{8}$$

where $\mathbf{R}_t$ is the world-frame torso orientation and ψ_{t-1} is its previous heading. At sequence initialization, the current heading is used. This convention removes absolute heading while retaining incremental turning information. The 6-D encoding is obtained from the first two columns of the rotation matrix. Global root translation and root linear velocity are not part of $\mathbf{x}_t$.

Proprioceptive history and action interface. The control decoder receives a 15-frame history of $\mathbf{s}_t^{\mathrm{pro}} = [{}^{\mathcal{B}}\boldsymbol{\omega}_t, \mathbf{q}_t - \mathbf{q}^{\mathrm{default}}, \dot{\mathbf{q}}_t, \mathbf{a}_{t-1}] \in \mathbb{R}^{90}$, where $\mathcal{B}$ denotes the base frame. The history is stored in term-major order, yielding $\mathbf{h}_t^{\mathrm{pro}} \in \mathbb{R}^{1350}$. Thus, encoder joint positions are distinguished from the default-pose-relative joint positions in the controller history, and torso angular velocity is distinguished from base angular velocity.

The policy outputs normalized actions rather than joint angles in radians. The PD targets are constructed as

$$\mathbf{q}_t^{\mathrm{des}} = \mathbf{q}^{\mathrm{default}} + \mathbf{s}_a \odot \mathbf{a}_t, \qquad \mathbf{a}_t \in \mathbb{R}^{29}, \tag{9}$$

where $\mathbf{s}_a$ is the joint-specific action scale. Joint-name mappings, default positions, and action scales are kept consistent across the encoder, controller, simulator, and hardware interfaces.

Teacher and control-native student architectures. The teacher follows the privileged tracking formulation of RoboGhost Li et al. (2026), adapted to the 29-DoF G1 interface. Its actor receives the current reference joint positions and velocities, anchor-pose, joint, and key-body tracking residuals, and proprioception. The asymmetric critic additionally receives privileged simulator states, including base linear velocity and rigid-body poses. The actor is a five-expert MoE with observation-dependent gating; the critic is an MLP.

The student jointly learns E_ϕ and D_{real} through control distillation. The encoder predicts a diagonal Gaussian posterior with $\boldsymbol{\mu}_t, \boldsymbol{\lambda}_t \in \mathbb{R}^{32}$, where $\boldsymbol{\lambda}_t$ is the log variance, clipped to $[-10, 4]$ for numerical stability. Student training uses reparameterized samples $\mathbf{z}_t$. The generator uses deterministic posterior means $\boldsymbol{\mu}_t$ as future supervision, while posterior samples appear in its historical condition through the execution-aware augmentation described below. Deployment uses the posterior mean to encode executed states. Table 6 summarizes the network architectures.

Normalization and latent interface. State and proprioceptive inputs use the frozen normalization statistics associated with the student checkpoint. Generator inputs and targets undergo a separate per-dimension standardization using training-split latent means and standard deviations. Generated latents are inverse-transformed before being passed to D_{real}. Measured states undergo the same encoder preprocessing and latent standardization when constructing generation history; these statistics are not updated online.

Table 6: **Policy architectures.** Hidden widths exclude the output layer.

Module	Input	Architecture	Output
Teacher actor	Reference, residuals, proprioception	5-expert MoE; $[512, 256, 128]$ per expert	29-D action
Teacher critic	Actor observations and privileged states	MLP $[512, 256, 128]$	Scalar value
Encoder E_ϕ	67-D motion state	MLP $[512, 512, 256]$; ELU; LayerNorm	32-D mean and log variance
Decoder D_{real}	32-D latent and 1350-D history	MLP with width-changing projections and same-width residual blocks; $[1024, 512, 512, 256, 128]$; ELU; LayerNorm	29-D action

Table 7: **Streaming generator architecture and diffusion settings.**

Setting	Value
Latent dimension	32
History / future length	20 / 40 frames
History / future duration	0.4 / 0.8 s
Text feature dimension	768
DiT width / blocks / heads	512 / 10 / 8
Feed-forward width / dropout	2048 / 0.1
Future position encoding	RoPE
Attention normalization	QK normalization
Training diffusion steps	1000
Prediction target / schedule	x_0 / cosine
Min-SNR parameter γ	5

Streaming latent DiT. The generator predicts 40 future 32-D latent frames from 20 history frames and a frozen Sentence-T5-XXL sentence embedding. The 768-D text feature is projected into the conditioning memory together with temporally encoded history tokens. Each DiT block applies self-attention among noisy future tokens and cross-attention to this memory, with diffusion-timestep modulation. Future self-attention is bidirectional within the predicted block; streaming is achieved through repeated conditional generation rather than a causal attention mask over future tokens. History-validity indicators distinguish padding from observations, and instruction-assignment indicators identify history frames associated with the current command.

A.2 Training Details

Retargeting and temporal alignment. We use HumanML3D and BABEL, including the BABEL-Stream organization of consecutive annotated actions. BABEL-Stream provides a streaming evaluation and training organization of BABEL rather than an independent motion source. Human motions are retargeted offline to G1 using GMR and resampled to 50 Hz. Joint positions and root translations are interpolated in time, root rotations use quaternion spherical linear interpolation, and velocities are recomputed from the resampled trajectories. Annotation boundaries are converted to the same timeline. This offline processing is distinct from online inference, which does not retarget explicit human motion.

Physical feasibility screening. Teacher-based screening first removes references that fail deterministic tracking under the teacher's termination checks, including excessive anchor-height, orientation, or selected body-position deviations. After student training, a separate MuJoCo pass screens candidate generator records using the frozen student driven by sampled posterior latents. This second pass rejects a record if torso height falls below 0.35 m, torso tilt exceeds 0.8 rad, or non-finite states occur. It does not apply a horizontal-displacement or keypoint-error cutoff. Together, the two screening stages improve the physical executability of the reference corpus.

The corpus-wide screening audit contains 150,552 candidate records, of which 128,486 are retained and 22,066 rejected. These counts include multiple source partitions and must not be interpreted as

Table 8: **Teacher tracking rewards.** Scales are the configured exponential-reward parameters; penalties do not use this parameter.

Term	Weight	Scale
Global anchor position	0.5	0.3
Global anchor orientation	0.5	0.4
Relative body position	1.0	0.3
Relative body orientation	1.0	0.4
Global body linear velocity	1.0	1.0
Global body angular velocity	1.0	3.14
Squared action change	-0.1	–
Joint-position limit violation	-10.0	–
Undesired contact	-0.1	–

Table 9: **Policy-training domain randomization.**

Quantity	Range
Static / dynamic friction	$\mathcal{U}(0.3, 1.6)$ / $\mathcal{U}(0.3, 1.2)$
Restitution	$\mathcal{U}(0, 0.5)$
Default joint-position offset	$\mathcal{U}(-0.01, 0.01)$ rad
Torso CoM offset, x	$\mathcal{U}(-0.025, 0.025)$ m
Torso CoM offset, y, z	$\mathcal{U}(-0.05, 0.05)$ m
Perturbation interval	$\mathcal{U}(1, 3)$ s
Linear-velocity perturbation, x, y	$\mathcal{U}(-0.5, 0.5)$ m/s
Linear-velocity perturbation, z	$\mathcal{U}(-0.2, 0.2)$ m/s
Angular-velocity perturbation, roll/pitch	$\mathcal{U}(-0.52, 0.52)$ rad/s
Angular-velocity perturbation, yaw	$\mathcal{U}(-0.78, 0.78)$ rad/s

the size of the generator training split. Generator optimization uses the designated training manifest. The evaluation sets are constructed independently of this student-screened subset; in particular, BABEL evaluation includes all 6,734 validation streams rather than only the streams retained by screening.

Simulation and reference-state initialization. Policies are trained in 4,096 parallel IsaacLab environments. A physics step of 0.005 s and control decimation of four yield a 50-Hz policy rate. Episodes initialize from sampled reference phases, setting the robot pose and velocities from the selected motion state and applying the configured reset perturbations. Training therefore covers intermediate motion states rather than only sequence onsets. Reference-state initialization is a training mechanism and is distinct from collecting measured history at deployment.

Teacher optimization and rewards. The teacher is trained with PPO using 24-step rollouts, five optimization epochs, four mini-batches, discount 0.99, GAE coefficient 0.95, clipping coefficient 0.2, entropy coefficient 0.005, and gradient clipping at 1. The initial learning rate is 10^{-3}, with KL-adaptive scheduling and a target KL of 0.01. The tracking configuration combines the reward terms in Table 8. Position and orientation terms use exponentially decaying tracking rewards; penalty terms discourage abrupt actions, joint-limit violations, and undesired contacts.

Domain randomization. Ground-contact parameters, default joint positions, and torso center of mass are randomized during policy training. External disturbances are applied by perturbing root velocities at random intervals. Table 9 reports the randomization ranges.

Student distillation. The student acts in simulation, and the frozen teacher labels the states visited by the student. The encoder processes the current reference motion state, while the control decoder additionally receives the student's actual proprioceptive history. Thus, the reference latent specifies the intended motion and the history describes the state from which it must be executed. Teacher-action supervision updates both the encoder and control decoder. At each iteration, at most 4,096 collected transitions are sampled for one optimization epoch with mini-batches of 512. We use AdamW with learning rate 10^{-4} and gradient clipping at 1. The objective is

$$\mathcal{L}_{\mathrm{VAE}} = \mathcal{L}_{\mathrm{ctrl}} + 10^{-4}\mathcal{L}_{\mathrm{KL}} + 10^{-5}\mathcal{L}_{z}, \tag{10}$$

where $\mathcal{L}_{\text{ctrl}}$ is the action Huber loss, $\mathcal{L}_{\text{KL}}$ is the posterior KL divergence to a unit Gaussian, and $\mathcal{L}_z$ is the mean squared sampled-latent magnitude. No motion reconstruction loss is included in this control-learning objective.

Teacher-free control ablation. The *w/o teacher* variant does not obtain a control objective by simply removing $\mathcal{L}_{\text{ctrl}}$. Instead, it keeps the same 67-D encoder, 32-D posterior, 1350-D proprioceptive history, 29-D control interface, motion data, rewards, and termination conditions, but learns E_ϕ and D_{real} directly from the motion-tracking return with PPO. During rollout, a base noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ is stored and reused when recomputing $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$ in the PPO update, so the policy-ratio computation is conditioned on the same latent sample and its gradient reaches both the encoder and control decoder. Each 24-step rollout is followed by five PPO epochs with four mini-batches. Thus, *w/o teacher* replaces action distillation with direct reinforcement learning under the matched architecture and tracking setup. In contrast, the main model is optimized by teacher-action distillation through Eq. equation 10, which contains no motion-reconstruction loss.

Latent extraction and execution-aware history augmentation. After student training, the encoder and controller are frozen. Reference motions are encoded into aligned posterior-mean and posterior-sample trajectories and standardized using the training-split latent statistics. The future supervision target always remains the deterministic posterior mean $\boldsymbol{\mu}$. Following Eq. equation 6, the historical condition is constructed as $\widetilde{\mathbf{H}}_t = (1-b)\mathbf{H}_t^{\mu} + b\mathbf{H}_t^{z} + \boldsymbol{\eta}$, where a single $b \sim \text{Bernoulli}(p_z)$ selects the latent source for each training window. We use $p_z = 0.30$ and $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, 0.01^2\mathbf{I})$ in standardized latent coordinates. The sampled history $\mathbf{H}_t^z$ is drawn from the posterior corresponding to the same reference frames; it therefore perturbs the conditioning interface without changing the supervision target. Source selection and Gaussian noise are applied only to valid history positions, leaving sequence-start padding and all future targets unchanged. The generator is trained offline on these augmented reference histories, not on simulator rollouts of its own predictions.

Training windows mix ordinary caption-balanced, instruction-boundary, and high-motion samples with probabilities 0.6, 0.2, and 0.2. The high-motion pool contains windows at or above the training-set 80th percentile of latent motion energy. This strategy increases coverage of command transitions and dynamic motions without changing the future prediction target.

Generator objective and optimization. We use a 1,000-step cosine diffusion process and direct x_0 prediction. The denoising MSE is weighted by $\min(\text{SNR}(\tau), 5)$. The full loss is

$$\mathcal{L}_{\text{gen}} = \mathcal{L}_{x_0} + 0.10\mathcal{L}_{\text{vel}} + 0.05\mathcal{L}_{\text{acc}} + 0.10\mathcal{L}_{\text{bnd}}. \tag{11}$$

The temporal losses match first- and second-order finite differences of predicted and target latents. The last valid clean history frame anchors these differences; when no history is valid, the first target frame is used during training. The boundary term supervises the first predicted latent and the first history-to-future difference. These are latent-space consistency terms, not physical joint-velocity or acceleration constraints.

The entire text condition is dropped with probability 0.2 for classifier-free guidance (CFG), while physical history is retained in both conditional and unconditional branches. Training settings are listed in Table 10; evaluation uses EMA weights.

A.3 Evaluation Details

Evaluation-only reconstruction. To measure how much motion information the control-trained latent retains, we freeze E_ϕ and fit a separate causal 1-D convolutional decoder D_{recon} on latent windows of length 16 to recover 67-D motion states using a Huber reconstruction objective. This decoder is an evaluation probe: its gradients do not update the encoder or control decoder, and it is not used for online action selection. The probe is matched to the latent convention under evaluation, distinguishing posterior means from samples. In the joint-versus-separate comparison, both frozen encoders are evaluated with the same probe architecture (hidden width 512, depth 3, and dilation growth 3), 16-frame context, batch size 512, learning rate 10^{-4}, and 50,000 optimization updates.

Reconstruction evaluation feeds encoded reference latents to this probe; generation evaluation feeds predicted latents. Consequently, motion-generation scores assess the generator together with its frozen motion readout, whereas physical execution tests whether the same latent interface can drive

Table 10: **Streaming generator training hyperparameters.**

Setting	Value
Optimizer / batch size	AdamW / 64
Optimization updates	2,000,000
Initial / minimum learning rate	10^{-4} / 10^{-6}
Warm-up / subsequent schedule	5,000 steps / cosine decay
Weight decay / gradient clipping	0.01 / 1.0
EMA decay	0.9999
Text dropout probability	0.20
Future latent target	Deterministic posterior mean $\boldsymbol{\mu}$
History base / sampled source	$\boldsymbol{\mu}$ / $\mathbf{z}$
History sampling probability p_z	0.30
History-noise s.d. σ_h	0.01
Ordinary / boundary / high-motion mixture	0.60 / 0.20 / 0.20
Training precision	bfloat16

D_{real}. For deterministic-$\boldsymbol{\mu}$ reconstruction, the motion is decoded once and the 20 evaluation repetitions resample retrieval groups and Diversity pairs. For sampled-$\mathbf{z}$ reconstruction, posterior noise is additionally resampled and the motion is decoded in every repetition. These repetitions estimate evaluation-time variability; they are not independent representation-training runs.

HumanML3D generation protocol. The test manifest contains 4,042 source motion clips. Retaining a full-motion caption where available and valid annotated temporal sub-clips yields 4,496 text–motion records. All methods are evaluated in the 67-D G1 representation at 50 Hz, using reference durations to separate motion quality from duration prediction. The evaluator applies common center cropping with a maximum length of 327 frames and the prescribed length-unit alignment. A single initial global heading is removed from each complete sequence, preserving relative turns. This evaluator convention differs from the frame-local heading convention used by the control encoder.

Methods marked with * are reproduced under the common robot-motion representation and evaluation split; their reported values are not imported from the original human-motion benchmarks. All methods share a frozen 67-D text–motion evaluator trained on HumanML3D training data. Retrieval grouping, Diversity-pair sampling, and sample ordering follow the same evaluation protocol for every method. Offline generation rolls forward predicted history because no physical robot is present; execution-state feedback is tested separately in simulation.

Reproduced baselines and comparison scope. The main generation table compares complete reproduced systems after mapping their outputs to the common 67-D G1 representation and evaluator. It is not a backbone-only comparison with identical language encoders, data filtering, or optimization budgets. Table 11 records the principal method-specific settings used in our reproductions. MotionStreamer and FloodDiffusion generate explicit robot motion and are paired with the same Humanoid-GPT tracker for physical execution. StaCo instead decodes its control-native latents directly with D_{real}.

The StaCo text-encoder ablation isolates Sentence-T5 versus CLIP within our architecture, but it does not attribute every cross-system difference in the main table to the generator backbone. The main table should therefore be read as a complete-system comparison under a unified output and evaluation interface; controlled component claims are drawn from the ablations.

Motion-generation metrics. Following MotionStreamer Xiao et al. (2025), FID compares Gaussian fits to real and generated motion embeddings, R@1/2/3 measures paired retrieval within groups of 32 candidates, and MM-Dist is the paired text–motion embedding distance. Diversity averages distances between 300 sampled motion pairs. Lower FID and MM-Dist and higher retrieval accuracy are preferred. For Diversity, agreement with the ground-truth reference is preferred over unbounded increases; the rightward arrow in the tables denotes this reference-matching criterion. These robot-domain scores are not directly comparable to scores obtained using human-motion representations or other evaluation encoders.

Reported generation scores are point estimates under this fixed protocol. Accordingly, small numerical changes across ablations are treated as descriptive rather than statistically significant; conclusions

Table 11: **Principal settings of reproduced generation systems.** Training recipes follow the respective methods, while output representation and evaluation are unified. "Final" and "best" denote checkpoint selection rather than an additional evaluation-time ensemble.

System	Training records	Text encoder	Optimization and selection	Inference
StaCo	Physically screened designated manifest	Sentence-T5-XXL	DiT: 2M updates, batch 64; final EMA	CFG 4; DDIM-20
MotionStreamer*	Common unfiltered set; 40,085 records	Sentence-T5-XXL	TAE: 2M updates; generator: 100k updates; effective batch 256; best validation loss	CFG 4
FloodDiffusion*	Same 40,085 training indices as MotionStreamer	UMT5-XXL	VAE: 300k updates; generator: 300k updates, batch 4; best EMA	CFG 4; 10 denoising steps
TextOp*	MVAE: 72,238 records; DAR: 40,047 records	CLIP ViT-B/32	MVAE/DAR: 200k/300k updates, batch 512; final	CFG 4; 8-frame primitives
DAJI*	DAJI-Act64 no-fall set; 94,860 records	Qwen3-VL-4B	DAJI-Flow: 200k updates, batch 16; final	No CFG; Euler-4

rely more on substantial changes or consistent trends across complementary metrics. The 20 reconstruction repetitions described above characterize metric and, where applicable, posterior-sampling variability, not uncertainty from retraining the generator.

BABEL long-term generation protocol. We evaluate all 6,734 BABEL-Stream validation pairs, comprising 13,468 annotated subsequences. The original two instructions, their change point, and the full sequence duration define each task. Each method generates a continuous stream across the instruction change rather than generating the two actions independently and concatenating them. This validation partition is the held-out BABEL benchmark used in our experiments.

Subsequence R@3, FID, Diversity, and MM-Dist are computed over the two annotated segments. Transition FID and Diversity use a one-second window centered on the instruction boundary, corresponding to 50 frames for the shared evaluator. Short boundary windows are padded by edge replication. The same frozen HumanML3D-trained evaluator is used without fitting a new BABEL evaluator.

Robot-domain PJ and AUJ. For transition smoothness, pose channels are resampled to 30 Hz and velocity channels are recomputed. In the centered 30-frame window, let

$$j_i(t) = \max_{d=1,\dots,29} |\Delta^3 q_{i,t,d}|, \qquad \bar{j}(t) = \frac{1}{N}\sum_{i=1}^{N} j_i(t). \tag{12}$$

We compute

$$\mathrm{PJ} = \max_t \bar{j}(t), \qquad \mathrm{AUJ} = \sum_t |\bar{j}(t) - \bar{j}_{\mathrm{GT}}(t)|. \tag{13}$$

The reduction first takes the largest absolute third difference across robot DoFs, then averages over sequences, and finally computes the peak or ground-truth discrepancy. No frame-rate-cubed or time-step factor is applied; these are discrete joint-angle smoothness statistics, not SI jerk values. The ground-truth jerk curve is recomputed in the robot domain, giving ground-truth AUJ exactly zero. PJ should approach its ground-truth value, whereas lower AUJ is preferred. The robot-domain adaptation must not be compared numerically with Cartesian human-joint PJ/AUJ values from other papers.

Reference tracking versus generated-motion execution. Reference tracking evaluates 4,042 HumanML3D motions in IsaacLab and MuJoCo. The encoder receives the retargeted reference state and the controller receives live proprioceptive history. End-to-end evaluation instead executes the text-conditioned command latents for the 4,496 text–motion records in MuJoCo. Its tracking target is each method's own generated motion, not the original HumanML3D trajectory. StaCo's decoded motion is used only as an evaluation reference; actions are still obtained directly from command latents.

MotionStreamer and FloodDiffusion are paired with Humanoid-GPT through the shared 67-D robot-motion interface. Since this representation omits global root translation and linear velocity, the tracker adapter reconstructs the required root commands without consulting test-reference root trajectories. Initial poses follow the generated first target in the tracking benchmark; neutral-standing startup is a separate deployment condition.

Tracking errors and alignment. The joint metric is the sequence-mean absolute error over 29 joint angles, reported in radians. Despite the inherited name E_{mpjpe}, it is not a Cartesian joint-position error. E_{mpkpe} averages Euclidean errors over the selected G1 key bodies and is reported in meters. In the generated-motion execution evaluator, reference keypoints are aligned to the robot's torso horizontal position and heading, while vertical discrepancies are retained. Thus, the metric does not assess global planar path accuracy. Each sequence contributes equally after its framewise errors are averaged.

Physical-execution metrics. We report completion-based success $\mathrm{Succ}_{\mathrm{nf}}$ and accuracy-aware success Succ. The latter requires physical stability throughout the sequence and sequence-mean $E_{\mathrm{mpjpe}} \leq 0.20$ rad and $E_{\mathrm{mpkpe}} \leq 0.15$ m. Success rates use the total number of evaluated sequences as their denominator. Tracking errors are first averaged over time within each trajectory and then averaged across the trajectories included in the corresponding benchmark, following the aggregation rules specified in the main experimental setup.

The reported 97.35% end-to-end success rate is defined specifically under this generated-motion execution protocol: the 4,496 trials are initialized from the first target pose of the corresponding generated sequence, each method is evaluated against its own generated target, and key-body errors are aligned in horizontal torso position and heading. It therefore measures stable and accurate execution of locally aligned generated poses, rather than global path following or the probability of completing an arbitrary language task from a neutral standing pose. Neutral-standing startup is evaluated only as a separate deployment condition.

A.4 Deployment Details

Zero-shot simulator transfer. The trained policies are transferred from IsaacLab to MuJoCo without fine-tuning. The transfer preserves joint ordering, action scaling, PD target construction, and the 50-Hz control interface. It tests robustness to simulator differences, but does not by itself establish robustness to every real-world contact or disturbance condition.

Closed-loop latent replanning. At each control step, measured robot states are converted into the 67-D representation and encoded as $\boldsymbol{\mu}_t^{\mathrm{exe}}$. The latest 20 execution latents and current text condition the prediction of 40 future command latents. After inverse normalization, each accepted command latent is decoded together with the continuously updated proprioceptive history. Only the executed state is appended to the feedback history; pending command latents remain in a separate plan buffer.

For the end-to-end simulation benchmark, we use fixed CFG scale 4 and 20-step DDIM with $\eta = 0$. Each plan contains 40 frames, and replanning occurs after 30 executed frames with five-frame linear overlap blending. A rolling noise buffer maintains temporal correspondence between overlapping plans. The deterministic DDIM update still depends on its initial diffusion noise. In the offline BABEL protocol, complete 40-frame plans are retained within an instruction, with five-frame blending when the instruction changes; this is distinct from execution-feedback replanning.

Asynchronous hardware execution. The hardware implementation separates a 50-Hz control loop from GPU planning. The frozen encoder and controller run through CPU ONNX inference, while the DiT runs in a separate GPU worker. The deployment configuration uses CFG scale 3 and DDIM-20, requests a new plan when ten frames remain, and blends five overlapping latent frames. The request threshold provides a 200-ms prefetch window; it is not a 200-ms per-action computation budget. Plans are aligned to the control timeline before acceptance.

On a text change, the system preserves measured history, updates its instruction-assignment indicators, and invalidates pending plans associated with older command versions. Text embeddings are cached and reused for replanning; previously unseen commands require an initial text-encoding pass. The first plan is prepared while the robot holds its initial posture and collects real history, rather than filling the history with an unexecuted generated motion.

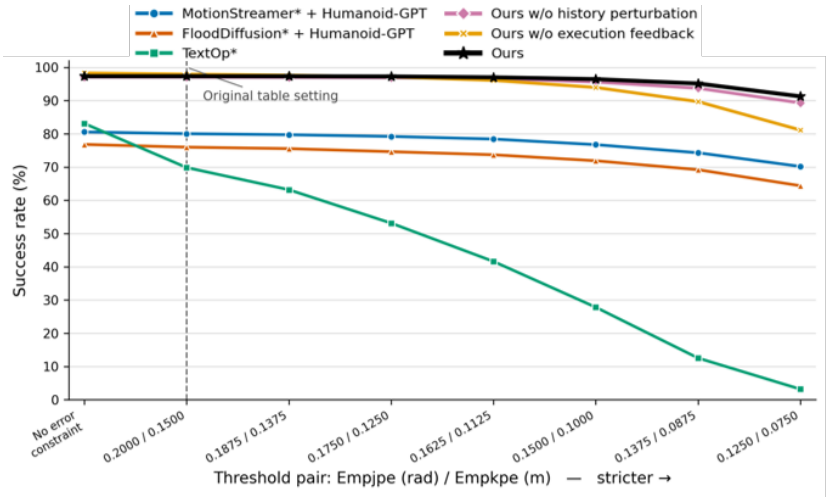

Figure 4: **Sensitivity to the success criterion.** End-to-end success recomputed under different joint-angle and key-body error thresholds using the same execution records and physical-failure flags. Execution feedback improves success under tighter precision requirements despite the slightly lower nominal success rate.

Real-robot state and command interface. The deployment interface derives torso orientation and angular velocity from the pelvis IMU and waist joint kinematics, then combines them with joint encoder measurements. It uses the Unitree SDK DDS interface to read robot state and send low-level joint commands. No privileged simulator state, motion-reconstruction decoder, or online human-motion retargeting is required. Stale-state, worker-health, and plan-buffer checks guard execution, with operator termination and fault handling returning the system to damping. These safeguards are engineering checks rather than a formal safety guarantee.

A.5 Additional Results

Sensitivity to the success criterion. Figure 4 recomputes success on the same execution records while varying the joint-angle and key-body thresholds. No model is retrained and no threshold-dependent trajectory subset is selected. At the nominal 0.20-rad/0.15-m criterion, the no-feedback variant has a slightly higher success rate. At 0.15 rad/0.10 m, the full model obtains 96.46% versus 93.97% without feedback; at 0.125 rad/0.075 m, the rates are 91.28% and 81.16%. Together with the lower tracking errors, these results indicate that execution-state feedback improves tracking precision, with a clearer advantage when tighter accuracy requirements are imposed.

Efficiency measurement. We measure batch-one inference on an NVIDIA GeForce RTX 5090 and an AMD Ryzen 9 9950X3D with 60.4 GiB system memory, using Python 3.11.15, PyTorch 2.7.0, CUDA 12.8, and cuDNN 9.7.1. Generation uses five warm-up calls and 50 timed calls; controller measurements use 100 warm-up calls and 2,000 timed calls. GPU measurements are synchronized. Text features are precomputed in a host cache, and request-time transfers and output transfers are included. Required neural motion readout is included for explicit-motion baselines; StaCo's state-encoder cost is charged to the generation side. Controller timing uses preassembled inputs. Model loading, text-cache construction, physics stepping, communication, and non-neural representation conversion are excluded.

We report amortized neural computation per executed frame:

$$\bar{T}_{\text{frame}} = \frac{\bar{T}_{\text{plan}}}{C} + \bar{T}_{\text{feedback}} + \bar{T}_{\text{ctrl}}, \tag{14}$$

where $C = 30$ for StaCo and $C = 40$ for the explicit-motion baselines. All methods generate 40 frames per call. The measured StaCo plan takes 72.22 ms on average, corresponding to 2.6963 ms/frame after including feedback encoding and control. Therefore, amortized cost must not be confused with single-plan latency or sensor-to-actuator delay.

Table 12: **Amortized neural computation at 50 Hz.** Generation is divided by the number of committed frames. The final column compares amortized mean and P95 costs with the 20-ms/frame throughput budget; it does not certify hard real-time scheduling.

Method	Gen.	Ctrl. (ms/frame)	Total	Within budget (20 ms/frame)
MotionStreamer* + Humanoid-GPT	12.3232	0.0995	12.4227	✓
FloodDiffusion* + Humanoid-GPT	6.3809	0.0995	6.4804	✓
Ours	2.4760	0.2203	2.6963	✓

The amortized P95 values are 2.7098, 12.4873, and 6.5442 ms/frame for StaCo, MotionStreamer, and FloodDiffusion, respectively. They are estimated from empirical timing-sample combinations, rather than measured as an end-to-end hardware latency trace. Costs below 20 ms/frame indicate sufficient measured neural throughput for 50-Hz streaming with buffering, not a hard real-time guarantee for every control tick. FloodDiffusion uses the available PyTorch scaled-dot-product attention backend in this environment.

Qualitative evidence and scope. The main-paper examples illustrate sequential commands and expressive whole-body behaviors, complementing the simulation benchmarks. Hardware examples are qualitative demonstrations, not estimates of population-level hardware success. The current representation does not specify global paths, scene geometry, or object interactions. Thus, the reported results primarily concern text-conditioned whole-body motion on the evaluated platform and conditions, rather than general scene-aware task execution.

A.6 Additional Ablation Details

Joint control learning versus separate representation learning. The separate-training baseline first learns a reconstruction-oriented motion representation and then freezes its encoder before training the latent-conditioned controller. Its representation stage uses the same 67-D-to-32-D encoder, frozen input normalization, sampled-$\mathbf{z}$ parameterization, KL weight 10^{-4}, and latent-magnitude weight 10^{-5} as StaCo. Its causal reconstruction decoder has width 512, depth 3, dilation growth 3, and a 16-frame context. This stage uses AdamW with learning rate 10^{-4} and batch size 512 for 416,000 optimizer updates.

After freezing this encoder, we train the same D_{real} architecture using the same teacher, motion manifest, DAgger rollout conditions, 4,096 environments, and sampled-latent convention as in StaCo. For representation evaluation, both the separately trained and control-trained frozen encoders are paired with the matched post-training probe specified above. These controls align encoder capacity, regularization, data, control training, and probe capacity/context within the corresponding stages, although the comparison is not intended as an equal-total-wall-clock compute study. The results therefore support the advantage of allowing teacher-action supervision to shape the encoder in this controlled implementation, rather than a universal claim over every possible separate representation-learning scheme. In all cases, the text-conditioned generator is trained only after the encoder and controller have been frozen.

Latent and generator variants. The latent-dimensionality comparison favors the 32-D representation over the more restrictive 8-D and 16-D bottlenecks. Removing teacher supervision preserves coarse stability but reduces precise-tracking performance, highlighting the role of action-level supervision. Sentence-T5 and mixed window sampling improve language alignment and generation quality within the tested configuration. The sampled-$\mathbf{z}$ generator is an independently trained ablation in which posterior samples define the generator's latent trajectories, including its prediction targets. This differs from the main model, which uses sampled $\mathbf{z}$ only as a stochastic historical condition and retains deterministic $\boldsymbol{\mu}$ as future supervision. Metric differences between these variants are reported descriptively and are not interpreted as statistically significant when the margin is small.

Feedback and history perturbation. The no-feedback variant uses previously generated command latents as generator history, while retaining live proprioceptive inputs to D_{real}. The full model instead conditions replanning on re-encoded execution states. Although the no-feedback variant attains a slightly higher nominal success rate, execution feedback reduces tracking errors and improves

success under tighter accuracy thresholds, consistent with better alignment between generated commands and physical execution. This ablation isolates the source of the generator-level history; it does not remove the low-level proprioceptive feedback used by the controller.

Both feedback variants use the proposed shared encoder interface. We have not included a matched generator conditioned directly on the raw 67-D state or on a separately trained state encoder. Consequently, this comparison supports the benefit of execution feedback within StaCo's shared control-latent interface, but does not by itself establish that this interface is necessary or optimal relative to those alternatives. The shared encoder is motivated by its consistent reference/execution representation and its direct compatibility with D_{real}; that design motivation is distinct from the empirical claim isolated by the no-feedback ablation.

Removing execution-aware history augmentation while retaining execution feedback produces only a small point difference, which we interpret as a modest trend rather than evidence of statistical significance. This ablation uses clean deterministic-$\boldsymbol{\mu}$ histories in place of the main model's posterior-sampled history substitution and additive Gaussian noise; future supervision remains deterministic $\boldsymbol{\mu}$ in both cases.

A.7 Algorithmic Summary

Training workflow.

1. Retarget and resample the motion corpus; train the privileged tracking teacher and screen references with deterministic teacher replay.
2. Roll out the student, query teacher actions at visited states, and jointly update E_ϕ and D_{real} with Eq. equation 10 using sampled latents.
3. Freeze the learned interface, screen generator records, and extract aligned posterior-mean and posterior-sample trajectories. Train the text-conditioned DiT with deterministic-$\boldsymbol{\mu}$ future targets, mixed historical latent sources, Gaussian history noise, and mixed window sampling. Fit the reconstruction probe separately for motion-space evaluation only.

Online execution workflow.

1. Read the robot state, construct $\mathbf{x}_t^{\mathrm{exe}}$, update the 15-frame proprioceptive history, and append the encoded execution mean to the 20-frame latent-history buffer.
2. When replanning is requested, snapshot the measured history and current instruction version, normalize the history, and generate a 40-frame plan in the planning worker.
3. Reject obsolete plans, align accepted plans with the control timeline, inverse-transform their latents, and blend overlapping frames.
4. Decode the next command latent with live proprioception, apply Eq. equation 9, and send the joint targets. Repeat while monitoring state freshness, buffer availability, and operator or fault-triggered termination.